```
# === INSTALL DEPENDENSI ===
!pip install -q 'git+https://github.com/facebookresearch/segment-anything.git'
!pip install -q roboflow supervision==0.23.0 shapely geopandas rasterio

# === UNDUH MODEL SAM ===
!mkdir -p path/to/weights
!wget -q https://dl.fbaipublicfiles.com/segment_anything/sam_vit_h_4b8939.pth -P
path/to/weights

# === INISIALISASI SAM ===
import os
import torch
from segment_anything import sam_model_registry, SamPredictor

CHECKPOINT_PATH = os.path.join("path/to/weights", "sam_vit_h_4b8939.pth")
print(CHECKPOINT_PATH, "; exist:", os.path.isfile(CHECKPOINT_PATH))

DEVICE = torch.device('cuda:0' if torch.cuda.is_available() else 'cpu')
MODEL_TYPE = "vit_h"

sam = sam_model_registry[MODEL_TYPE](checkpoint=CHECKPOINT_PATH).to(device=DEVICE)
predictor = SamPredictor(sam)

# === IMPORT PUSTAKA UMUM ===
import cv2
import numpy as np
import geopandas as gpd
import rasterio
from shapely.geometry import Polygon
import shapely.affinity
from tqdm import tqdm
import pandas as pd

# === ROOT FOLDER INPUT/OUTPUT ===
ROOT_FOLDER = "path/to/output_pipeline"
RESOLUTION_FOLDERS = ["1280", "1920", "2560"]

# === FUNGSI: Konversi mask biner ke daftar polygon ===
def mask_to_polygons(mask):
    contours, _ = cv2.findContours(mask.astype(np.uint8), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
    polygons = []
    for contour in contours:
        if len(contour) >= 3:
            poly = Polygon(contour.squeeze())
            if poly.is_valid and poly.area > 0:
                polygons.append(poly)
    return polygons

# === FUNGSI: Gabungkan seluruh GeoJSON dalam folder menjadi satu file ===
```

```
def merge_geojson_files(folder_path, output_path):
    geojson_files = [f for f in os.listdir(folder_path) if f.endswith(".geojson")]
    merged_gdf = gpd.GeoDataFrame()
    for file in geojson_files:
        gdf = gpd.read_file(os.path.join(folder_path, file))
        merged_gdf = pd.concat([merged_gdf, gdf], ignore_index=True)
    if not merged_gdf.empty:
        merged_gdf.to_file(output_path, driver="GeoJSON")

# === LOOP PER FOLDER RESOLUSI ===
for res_folder in RESOLUTION_FOLDERS:
    base_path = os.path.join(ROOT_FOLDER, res_folder)
    txt_path = os.path.join(base_path, "txt_bbox")
    jpg_path = os.path.join(base_path, "jpg_tiles")
    tif_path = os.path.join(base_path, "tif_tiles")
    mask_output_path = os.path.join(base_path, "geojson_mask")
    bbox_output_path = os.path.join(base_path, "geojson_bbox")

    os.makedirs(mask_output_path, exist_ok=True)
    print(f"Memproses folder: {res_folder}")

    for file in tqdm(sorted(os.listdir(txt_path))):
        if not file.endswith(".txt"):
            continue

        tile_name = os.path.splitext(file)[0]
        txt_file = os.path.join(txt_path, f"{tile_name}.txt")
        jpg_file = os.path.join(jpg_path, f"{tile_name}.jpg")
        tif_file = os.path.join(tif_path, f"{tile_name}.tif")

        if not (os.path.exists(jpg_file) and os.path.exists(tif_file)):
            print(f"Lewatkan {tile_name} (jpg atau tif tidak ditemukan)")
            continue

        # Baca dan atur image untuk SAM
        image_bgr = cv2.imread(jpg_file)
        image_rgb = cv2.cvtColor(image_bgr, cv2.COLOR_BGR2RGB)
        predictor.set_image(image_rgb)

        # Ambil transformasi dan CRS dari GeoTIFF
        with rasterio.open(tif_file) as src:
            transform = src.transform
            crs = src.crs

        # Baca deteksi bounding box dari YOLO
        with open(txt_file, "r") as f:
            lines = f.readlines()

        features = []
```

```
        for line in lines:
            parts = line.strip().split()
            if len(parts) != 6:
                continue

            class_name, conf = parts[0], float(parts[1])
            x_min, y_min, x_max, y_max = map(float, parts[2:])
            box = np.array([x_min, y_min, x_max, y_max])

            try:
                masks, scores, _ = predictor.predict(box=box,
multimask_output=True)
                best_index = np.argmax(scores)
                best_mask = masks[best_index]
                best_score = float(scores[best_index])
            except Exception as e:
                print(f"Gagal prediksi mask di {tile_name} - {e}")
                continue

            polygons = mask_to_polygons(best_mask)
            for poly in polygons:
                geo_poly = shapely.affinity.affine_transform(
                    poly,
                    [transform.a, transform.b, transform.d, transform.e,
transform.xoff, transform.yoff]
                )
                features.append({
                    "geometry": geo_poly,
                    "properties": {
                        "class_name": class_name,
                        "bbox_confidence": conf,
                        "mask_score": best_score
                    }
                })

        # Simpan hasil mask sebagai GeoJSON
        if features:
            gdf = gpd.GeoDataFrame.from_features(features, crs=crs)
            output_geojson = os.path.join(mask_output_path, f"{tile_name}.geojson")
            gdf.to_file(output_geojson, driver="GeoJSON")

    # === GABUNGKAN FILE GEOJSON ===
    print(f"Menggabungkan file GeoJSON di folder {res_folder}")
    merge_geojson_files(
        folder_path=bbox_output_path,
        output_path=os.path.join(base_path, f"{res_folder}_bbox_merge.geojson")
    )
    merge_geojson_files(
        folder_path=mask_output_path,
        output_path=os.path.join(base_path, f"{res_folder}_mask_merge.geojson")
```

)